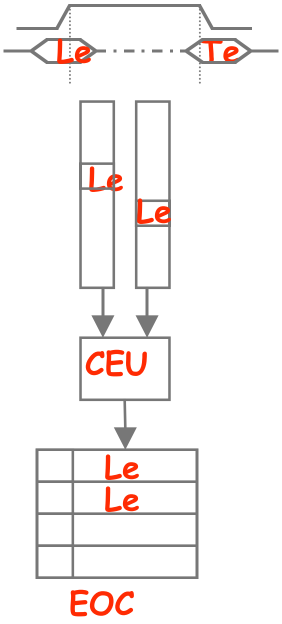
# RCU: MAIN SEQUENCER

R. Marchesini / LBL Berkeley
E-Mail: RMarchesini@lbl.gov

- Outline:
- Parallel Operations of the Read Out
- Control Signals of the main blocks and handshakes.
  - EOC
  - FIFO
  - SERIALIZER
- Simulations

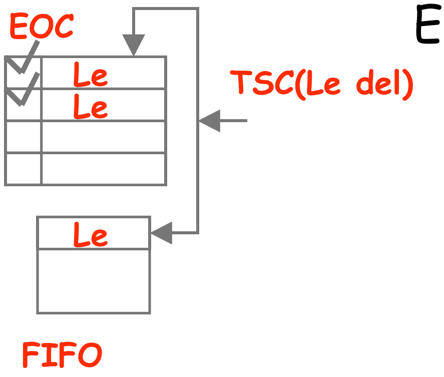
# Parallel Operations

Three main operations are performed in parallel:



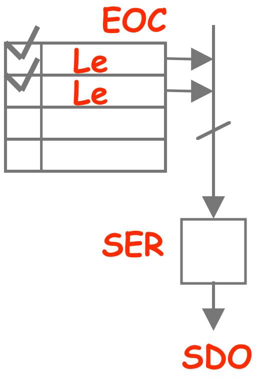
- Column Read Out and data storage in the EOC:

  The data is stored in the pixel memory and the CEU stores it in sequence into the EOC. Each data consist of  LE, TE and ADD information.



- Data selection for read out {Le comparison}

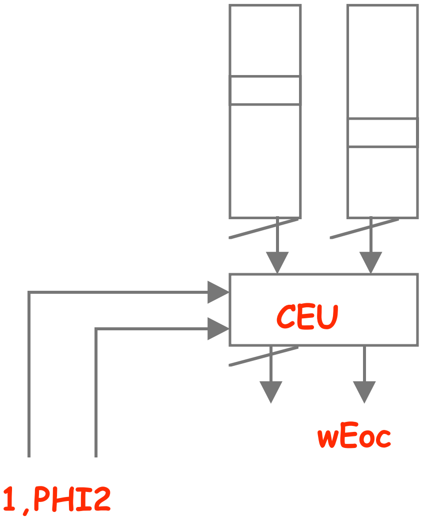
  Each trigger is assigned to a time stamp which is the same used for the pixel but delayed by the latency. This Trigger time is stored into the FIFO and used also to select the data in the EOC that matches the same time (LE comp.)



- Read out sequence

  For each trigger a read out sequence is performed.  This sequence reads out the selected data in the EOC and serializes it out.

# Column & CEU

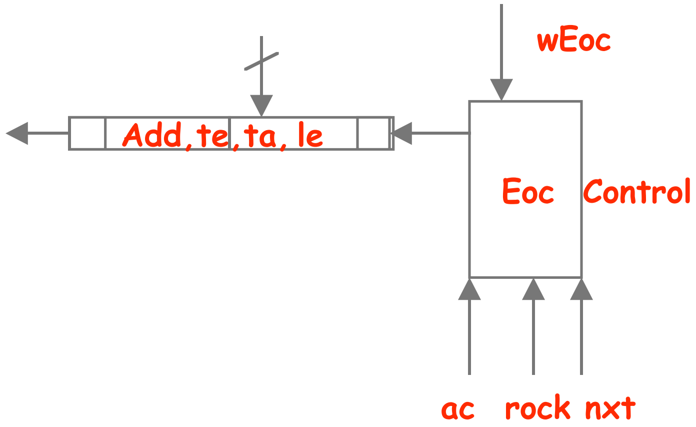


The CEU manages two columns reading the data stored into the pixel memory, and write it into the Eoc.

- Operations:
  - 1. Hit detection
  - 2. Sparse scan
  - 3. Strobe for writing the data in the EOC.

When the discriminator fire the LE and TE are stored in the pixel memory, and a priority signal ripple down, initiating the data read out. The read out strobe are generated with a 2 phase clock (PHI1, PHI2).

# EOC



- Operations:
  - 1) WRITE, Write pixel data (LE, TE, ADD).
  - 2) LE COMPARISON, Compare the Trigger time stamp (LE delayed) and store the trigger # (TAI)

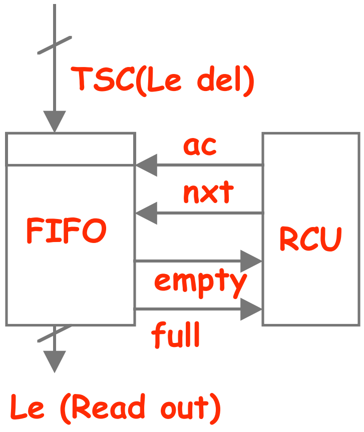
  3-4) READ OUT, read out all the data that matches the actual trigger # (TeEoc, AddEoc).

  XCK -> 40 MHz, RCK -> 10 MHz

| Op. | Signals | From |
|---|---|---|
| 1) | wEoc ( ^PHI1) | CEU |
| 2) | ac (^TRIGGER^XCK) | RCU |
| 3) | nxt (^RCK ^XCK) | RCU |
| 4) | rock (^RCK ^XCK) | RCU |
| | rock9 | Horiz.Scan |
| | prioO | Horiz..Scan |

Handshake: no rock are produced when all the event are read out (Rock9 -> 1) EOE .

# FIFO



The main function is to store the LE delayed by the latency every time there is a trigger, and retrieve the LE information during the read out phase.

- Operations:
  - 1) WRITE.
  - 2) READ.

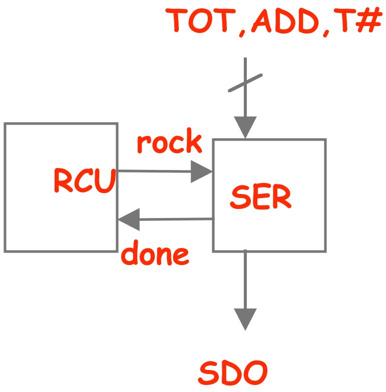
Two independent ports for writing and reading.

| Op. | Signal | From: |
|---|---|---|
| 1) Write | acFifo | RCU |
|  | full | FIFO |
|  | empty | FIFO |
| 2) Read | nxt | RCU |

Handshake:

A nxt is produced when a read out sequence is terminated and the fifo is not empty,

# Serializer



The rock transfer the data into the serializer.

| Op. | Signal | | From |
|---|---|---|---|
| SDO | rock | (^RCK) | RCU |
| | done | | SER. |

Handshake:

No rock is issued before the done is produced.

# Simulation

Actual simulation is performed on a single column pair and whole read out logic. The actual model for the different block is the following:

RCU: gate level simulation with typical delays.

FIFO: no delays, most gate level, functional view of the memory cell.

SERIALIZER: gate level.

SUBTRACTOR: behavioral model, no delays.

COUNTER: behavioral view, indicative delays.

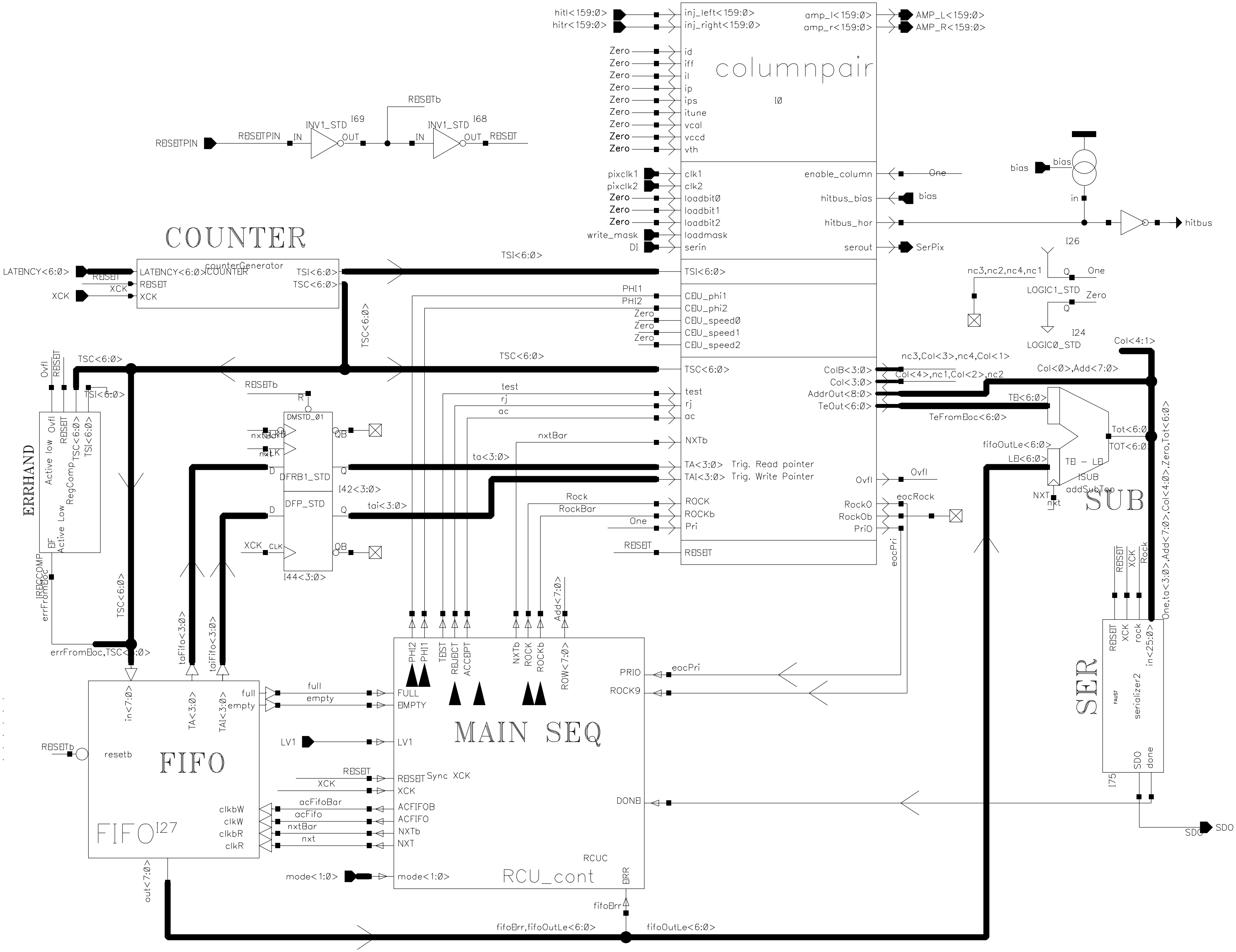
COLUMN and CEU: low level behavioral model, gate level. All with typical delay.

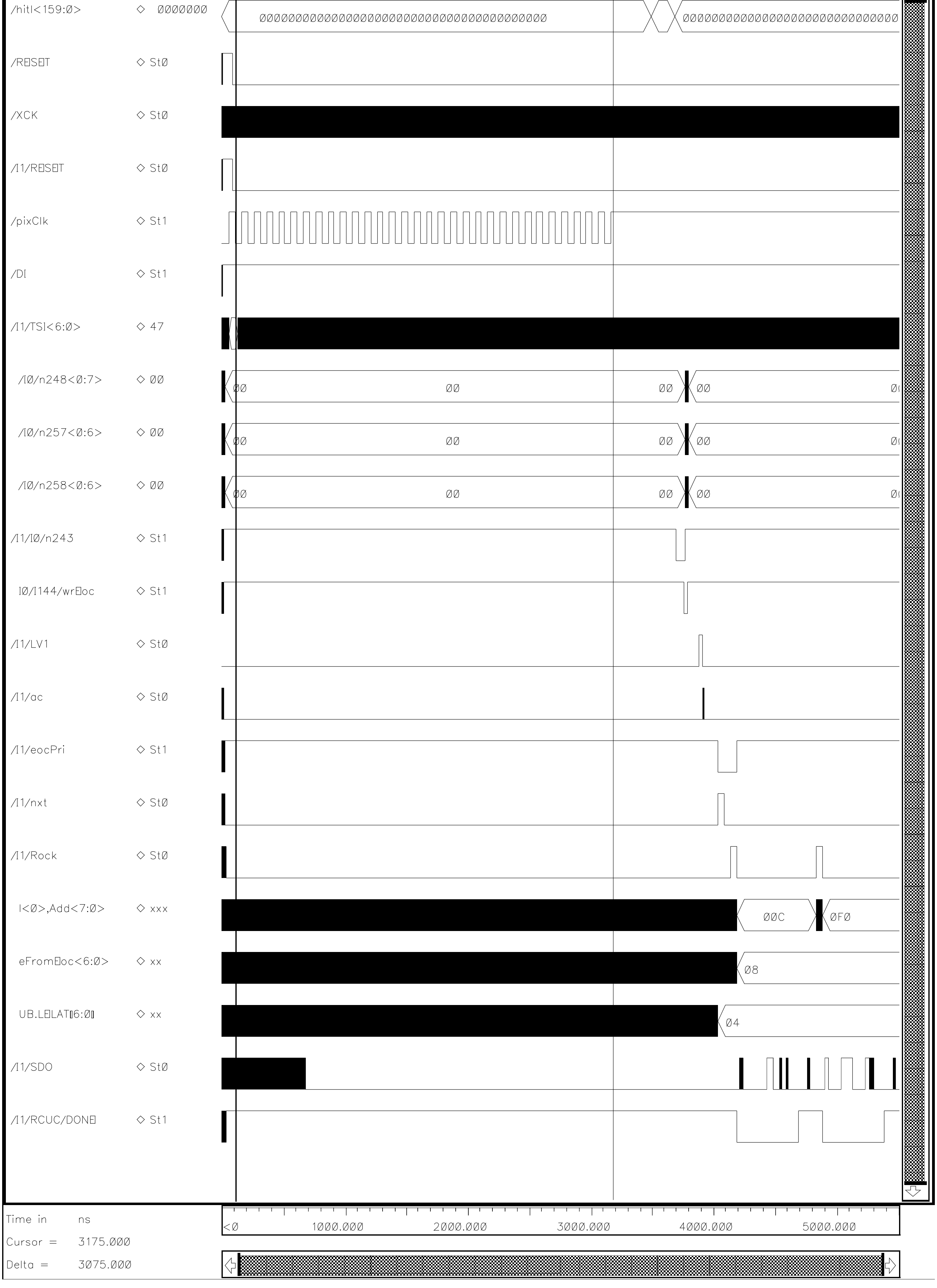
EOC: gate level and memory cell behavioral model, no delays.

Next: model all the blocks at the gate level when it's possible, and annotate the delays from spice simulation.

The verilog simulation will be performed on the whole chip circuit. It is mainly intended to be used to stress the logic test, covering all the non standard chip operation. This will also check the connectivity between large blocks.

Spice simulation will be performed on large parts, ex: 2 columns pairs and the whole read out logic.

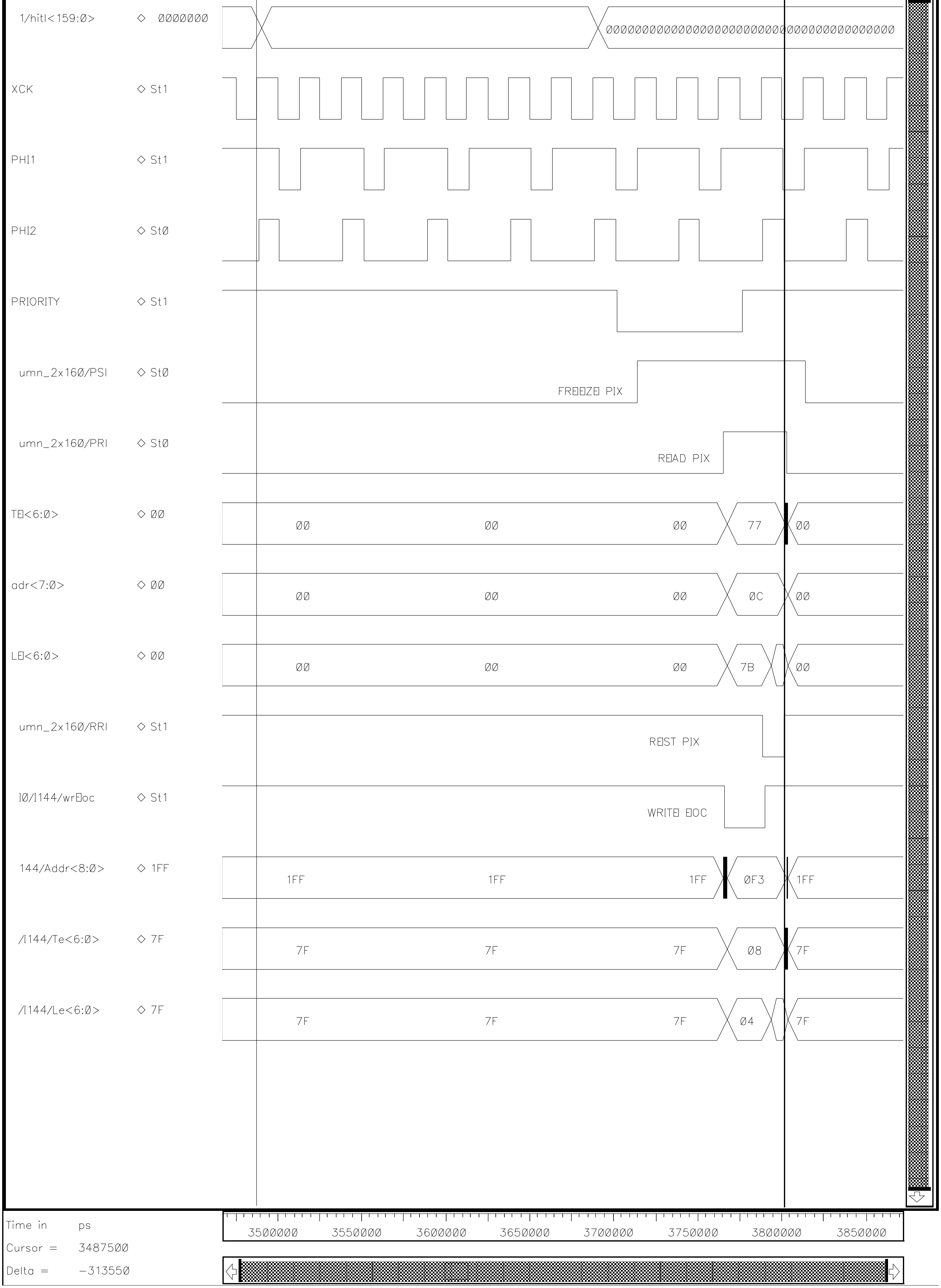

columnpair
I0
hitl<159:0>
hitr<159:0>
inj_left<159:0>
inj_right<159:0>
amp_l<159:0>
amp_r<159:0>
AMP_L<159:0>
AMP_R<159:0>
Zero
id
iff
il
ip
ips
itune
vcal
vccd
vth
pixclk1
pixclk2
clk1
clk2
loadbit0
loadbit1
loadbit2
loadmask
serin
write_mask
DI
enable_column
One
hitbus_bias
bias
hitbus_hor
hitbus
serout
SerPix
RESETb
RESETPIN
RESET
INV1_STD
I69
I68
IN
OUT
COUNTER
counterGenerator
ICOUNTER
LATENCY<6:0>
TSI<6:0>
TSC<6:0>
XCK
PHI1
PHI2
CEU_phi1
CEU_phi2
CEU_speed0
CEU_speed1
CEU_speed2
I26
nc3,nc2,nc4,nc1
Q
LOGIC1_STD
LOGIC0_STD
I24
Col<4:1>
nc3,Col<3>,nc4,Col<1>
Col<4>,nc1,Col<2>,nc2
Col<0>,Add<7:0>
ColB<3:0>
Col<3:0>
AddrOut<8:0>
TeOut<6:0>
TE<6:0>
TeFromEoc<6:0>
Tot<6:0>
TOT<6:0>
fifoOutLe<6:0>
LE<6:0>
TE - LE
ISUB
addSubTop
SUB
NXT
nxt
test
rj
ac
nxtBar
NXTb
ta<3:0>
TA<3:0> Trig. Read pointer
TAI<3:0> Trig. Write Pointer
Ovfl
Rock
RockBar
ROCK
ROCKb
Pri
RockO
RockOb
PriO
eocRock
eocPri
ERRHAND
Active low
Active Low
RegComp
EF
errFromEoc
ERRCOMP
DMSTD_01
DFRB1_STD
DFP_STD
I42<3:0>
I44<3:0>
CLK
QB
D
R
tai<3:0>
taFifo<3:0>
taiFifo<3:0>
errFromEoc,TSC<6:0>
Add<7:0>
PHI2
PHI1
TEST
REJECT
ACCEPT
ROW<7:0>
PRIO
ROCK9
FULL
EMPTY
full
empty
LV1
Sync XCK
ACFIFOB
ACFIFO
NXT
acFifoBar
acFifo
nxtBar
nxt
MAIN SEQ
RCUC
RCU_cont
mode<1:0>
ERR
DONE
fifoErr
fifoErr,fifoOutLe<6:0>
FIFO
FIFO I27
in<7:0>
TA<3:0>
TAI<3:0>
resetb
clkbW
clkW
clkbR
clkR
out<7:0>
SER
serializer2
FAUST
I75
rock
in<25:0>
One,ta<3:0>,Add<7:0>,Col<4:0>,Zero,Tot<6:0>
SDO
done
Rock



```
/hitl<159:Ø>          ◇  ØØØØØØØ
/R□SE□T               ◇ StØ
/XCK                  ◇ StØ
/I1/R□SE□T            ◇ StØ
/pixClk               ◇ St1
/DI                   ◇ St1
/I1/TSI<6:Ø>          ◇ 47
 /IØ/n248<Ø:7>        ◇ ØØ
 /IØ/n257<Ø:6>        ◇ ØØ
 /IØ/n258<Ø:6>        ◇ ØØ
/I1/IØ/n243           ◇ St1
 IØ/I144/wr□oc        ◇ St1
/I1/LV1               ◇ StØ
/I1/ac                ◇ StØ
/I1/eocPri            ◇ St1
/I1/nxt               ◇ StØ
/I1/Rock              ◇ StØ
 I<Ø>,Add<7:Ø>        ◇ xxx
 eFrom□oc<6:Ø>        ◇ xx
 UB.L□LAT▯6:Ø▯        ◇ xx
/I1/SDO               ◇ StØ
/I1/RCUC/DONE□        ◇ St1
```

ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ
ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØ
ØØC
ØFØ
Ø8
Ø4
<Ø
1ØØØ.ØØØ
2ØØØ.ØØØ
3ØØØ.ØØØ
4ØØØ.ØØØ
5ØØØ.ØØØ

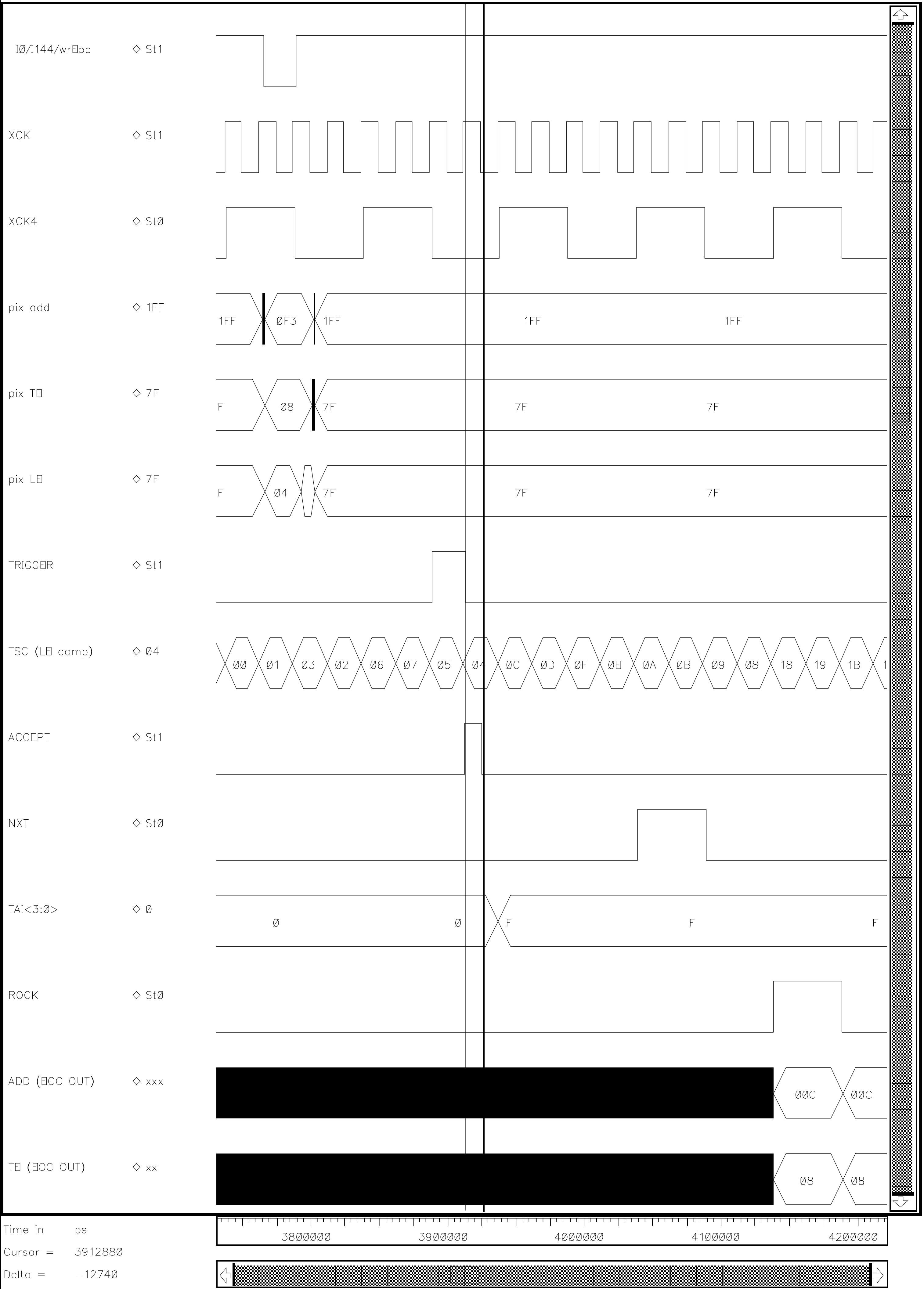
```
Time in     ns
Cursor =    3175.ØØØ
Delta =     3Ø75.ØØØ
```



Time in ps

Cursor = 3487500

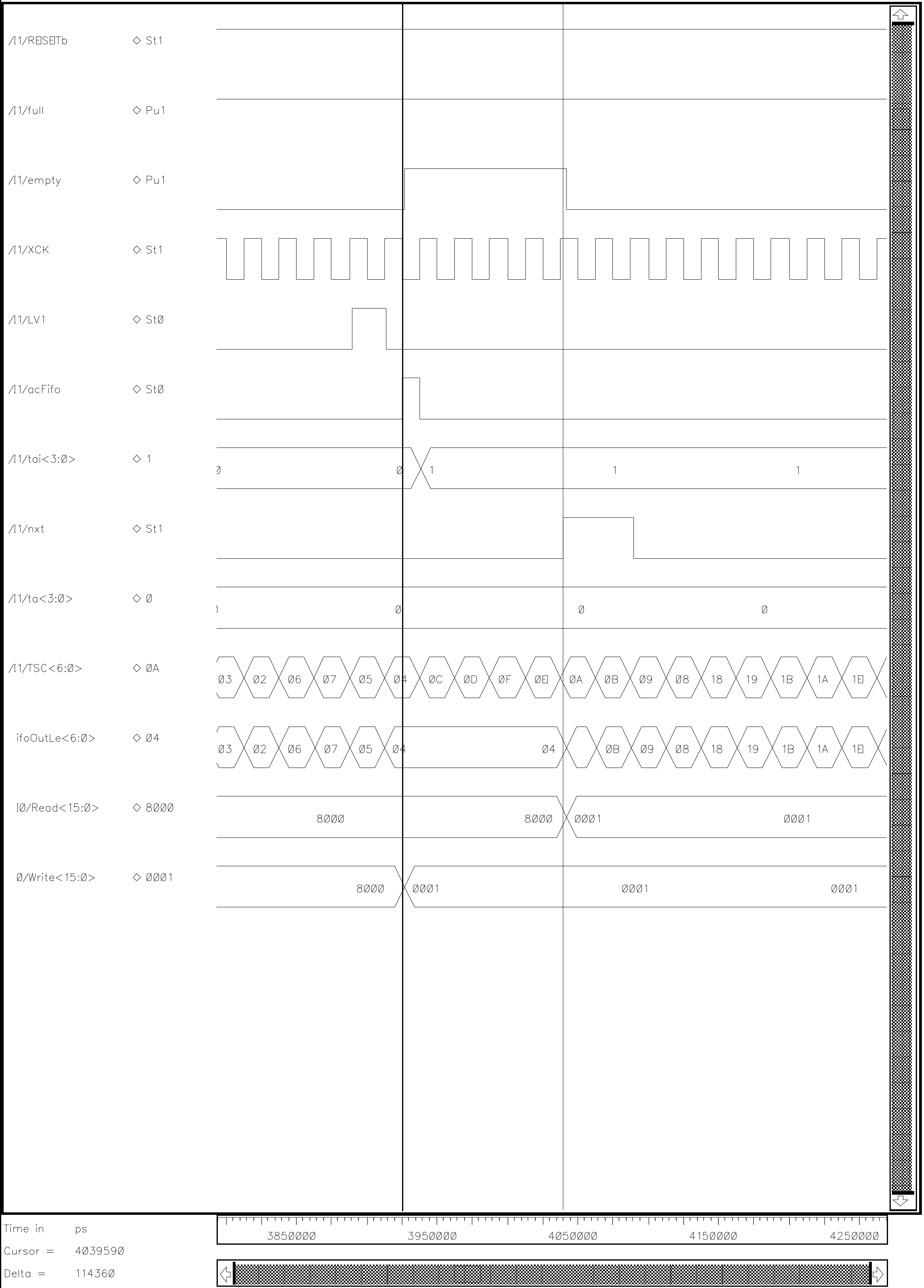
Delta = −313550

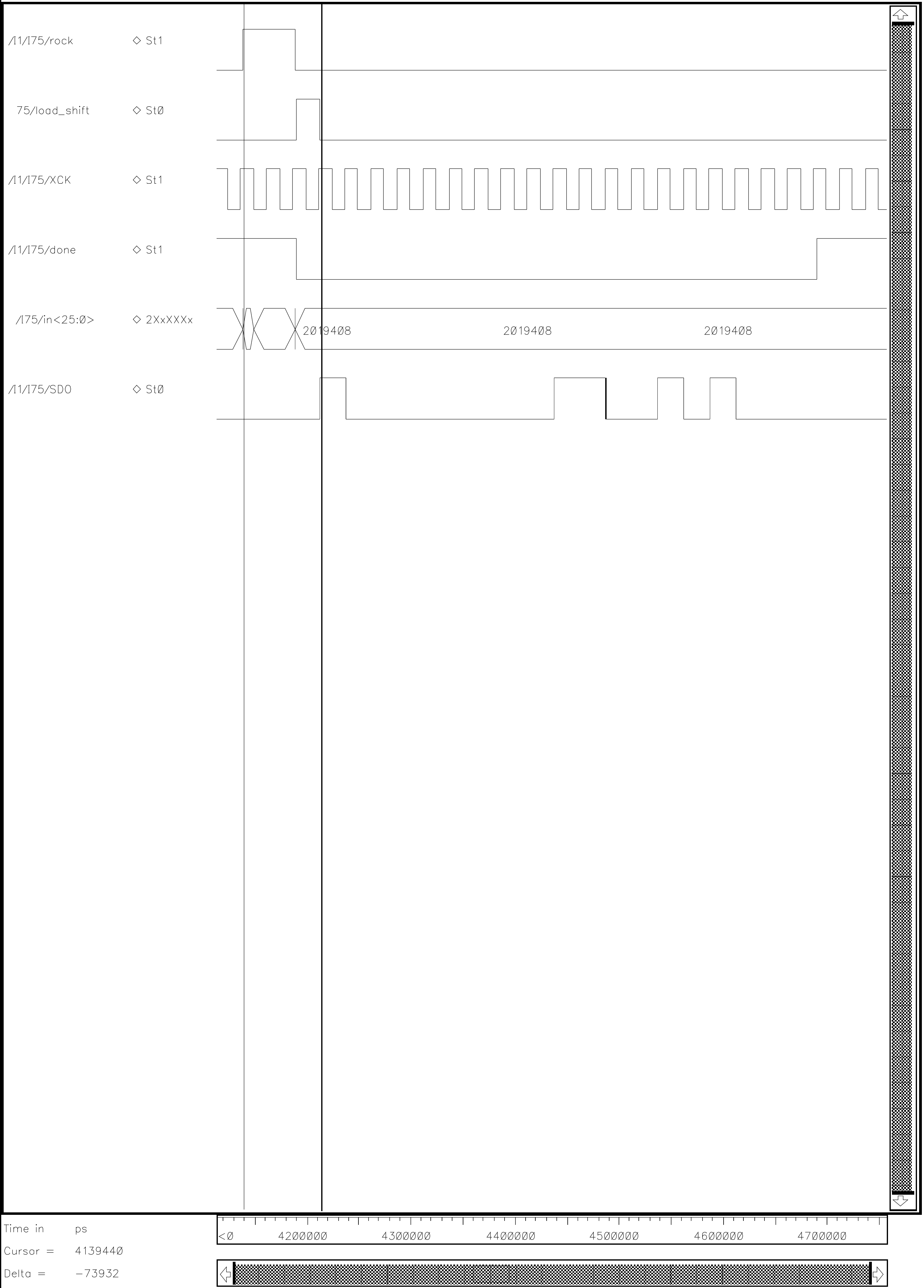


Time in ps

Cursor = 3912880

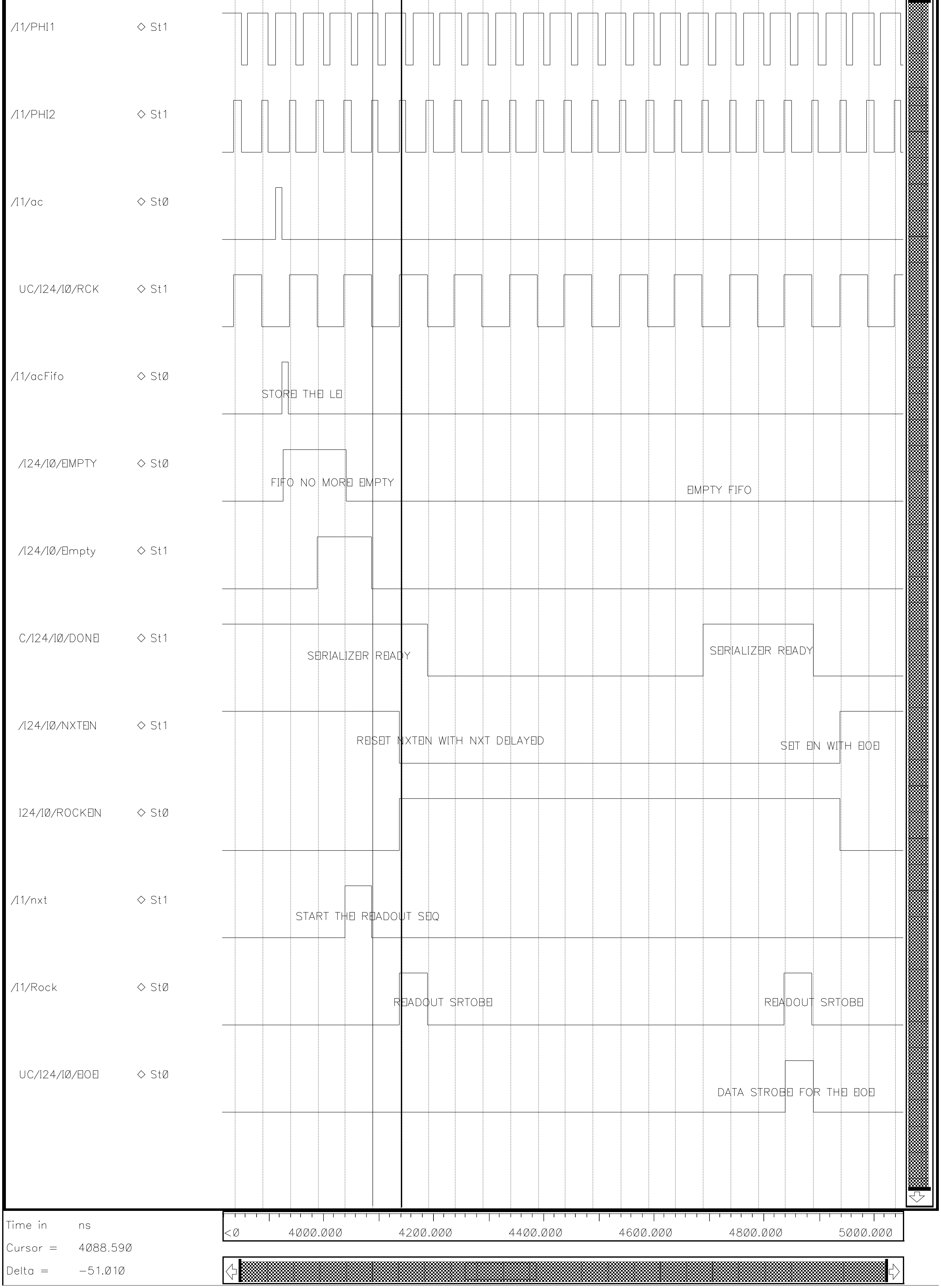
Delta = −12740


/I1/RESETb ◇ St1
/I1/full ◇ Pu1
/I1/empty ◇ Pu1
/I1/XCK ◇ St1
/I1/LV1 ◇ StØ
/I1/acFifo ◇ StØ
/I1/tai<3:Ø> ◇ 1
Ø Ø 1 1 1
/I1/nxt ◇ St1
/I1/ta<3:Ø> ◇ Ø
Ø Ø Ø
/I1/TSC<6:Ø> ◇ ØA
Ø3 Ø2 Ø6 Ø7 Ø5 Ø4 ØC ØD ØF ØE ØA ØB Ø9 Ø8 18 19 1B 1A 1E
ifoOutLe<6:Ø> ◇ Ø4
Ø3 Ø2 Ø6 Ø7 Ø5 Ø4 Ø4 ØB Ø9 Ø8 18 19 1B 1A 1E
IØ/Read<15:Ø> ◇ 8ØØØ
8ØØØ 8ØØØ ØØØ1 ØØØ1
Ø/Write<15:Ø> ◇ ØØØ1
8ØØØ ØØØ1 ØØØ1 ØØØ1
Time in ps
Cursor = 4Ø3959Ø
Delta = 11436Ø
385ØØØØ
395ØØØØ
4Ø5ØØØØ
415ØØØØ
425ØØØØ

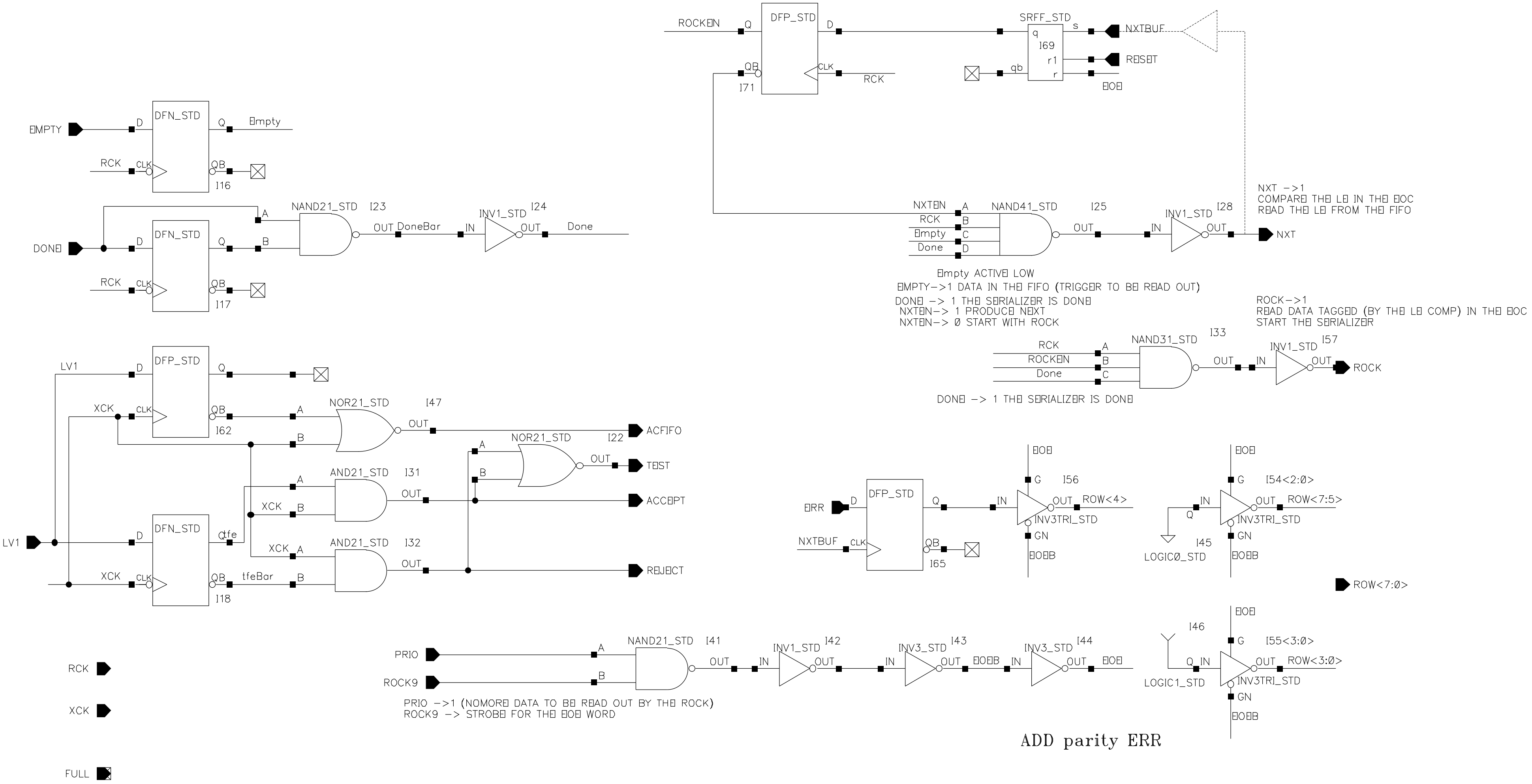


```
/I1/I75/rock        ◇ St1
 75/load_shift      ◇ StØ
/I1/I75/XCK         ◇ St1
/I1/I75/done        ◇ St1
 /I75/in<25:Ø>      ◇ 2XxXXXx
/I1/I75/SDO         ◇ StØ

Time in    ps
Cursor =   413944Ø
Delta =    -73932
```



Time in ns
Cursor = 4088.590
Delta = -51.010



NXT ->1
COMPARE THE LE IN THE EOC
READ THE LE FROM THE FIFO

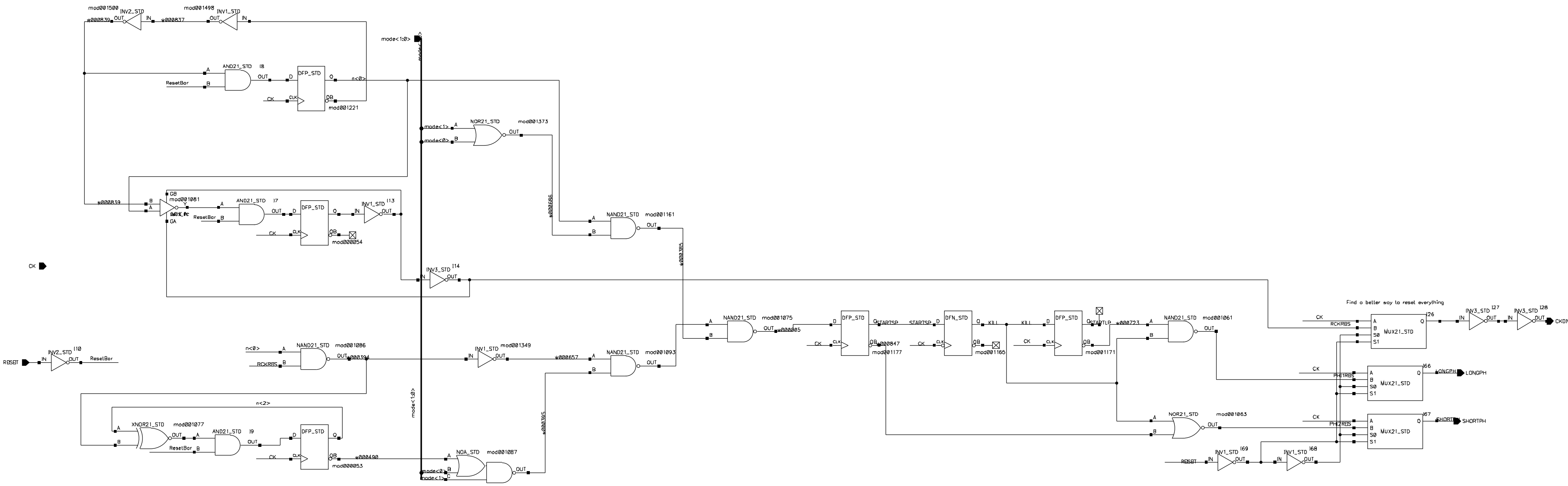
Empty ACTIVE LOW
EMPTY->1 DATA IN THE FIFO (TRIGGER TO BE READ OUT)
DONE -> 1 THE SERIALIZER IS DONE
NXTEN-> 1 PRODUCE NEXT
NXTEN-> 0 START WITH ROCK

ROCK->1
READ DATA TAGGED (BY THE LE COMP) IN THE EOC
START THE SERIALIZER

DONE -> 1 THE SERIALIZER IS DONE

PRIO ->1 (NOMORE DATA TO BE READ OUT BY THE ROCK)
ROCK9 -> STROBE FOR THE EOE WORD

ADD parity ERR



mod001500
INV2_STD
w000839
OUT
IN
w000837
mod001498
INV1_STD
OUT
IN
mode<1:0>
mode<1:0>
AND21_STD
I8
A
B
ResetBar
OUT
D
DFP_STD
Q
n<0>
CK
CLK
QB
mod001221
mode<1>
A
mode<0>
B
NOR21_STD
mod001373
OUT
w000686
NAND21_STD
mod001161
A
B
OUT
w000385
w000839
B
A
GB
mod001081
Y
MUX_PC
GA
AND21_STD
I7
ResetBar
OUT
DFP_STD
Q
QB
mod000054
INV1_STD
I13
IN
OUT
CK
INV3_STD
I14
IN
OUT
RESET
INV2_STD
I10
IN
OUT
ResetBar
n<0>
RCKRBS
NAND21_STD
mod001086
OUT
w000394
INV1_STD
mod001349
IN
OUT
w000657
NAND21_STD
mod001093
A
B
OUT
n<2>
XNOR21_STD
mod001077
OUT
AND21_STD
I9
ResetBar
DFP_STD
Q
QB
w000490
mod000053
NOA_STD
mod001087
mode<0>
mode<1>
C
OUT
w000395
NAND21_STD
mod001075
OUT
w000005
DFP_STD
STARTSP
QB
w000847
mod001177
STARTSP
DFN_STD
KILL
mod001165
KILL
DFP_STD
STARTLP
w000723
mod001171
NAND21_STD
mod001061
OUT
NOR21_STD
mod001063
OUT
Find a better way to reset everything
CK
RCKRBS
MUX21_STD
I26
A
B
S0
S1
Q
INV3_STD
I27
IN
OUT
INV3_STD
I28
IN
OUT
CKDN
CK
PHI1RBS
MUX21_STD
I66
LONGPH
LONGPH
CK
PHI2RBS
MUX21_STD
I67
SHORTPH
SHORTPH
RESET
INV1_STD
I69
IN
OUT
INV1_STD
I68
IN
OUT